What To Expect When You're Expecting: On the Creative Potential of Generative AI

Tony Veale

1. Introduction

If a good story is not unlike a well-written computer program, the similarities run deeper than the fact that each is, ultimately, a kind of text. Rather, a story is a kind of literary software for achieving a desired effect on a reader. While some stories thrill, others chill, and some make us laugh or cry. As texts, both stories and programs concern themselves with the efficient flow of information, so each emerges as a surface rendering of a deeper logical structure: for stories, we call this structure the *fabula* or *plot* (Pérez y Pérez and Sharples 2023); for a computer program, this structure is the underlying algorithm. Moreover, writers of both kinds of software must be attentive to the limitations of their target hardware, from the size of one's memory to one's capacity for parallel and recursive processing. Given these similarities, it is hardly surprising that even before the advent of modern software, creators had sought to systematize the engineering of new stories along similar lines.

Certainly, the large language models (or LLMs) that have effected a recent and dramatic sea-change in the public perception of Artificial Intelligence (AI) see little difference between stories – or literature of any kind – and computer programs. To an LLM, a text is just a text, a contiguous sequence of tokens that is *auto-regressively* generated one token at a time (Radford et al., 2019; Bender et al., 2021). This makes LLMs just as adept at authoring programs as stories. What elevates an LLM above the level of digital philistine is the depth and complexity of its neural architecture, which turns each model into a heavily trained, finely tuned and highly contextualized player of language games. The writing of a story involves very different language games than the writing of a program, a poem, a joke, or a shopping list, yet while the rules may differ, the process remains the same, to LLMs at least. What changes is the context to which the LLM is attuned via its complex web of weights and probabilities.

The dramatic success of LLMs represents a victory for the neural/statistical paradigm over the purely symbolic alternative, but this victory was not always assured. Despite showing some initial promise, research in artificial neural networks (ANNs) foundered during the first *AI Winter* of the 1970s, as early ANNs proved difficult to scale and to train. Before these barriers could be overcome, the impetus shifted back to the symbolic paradigm, which was, after all, the paradigm in which scholars had first made major inroads into the structural understanding of stories. But the symbolic paradigm would endure its own AI winter, before the advent of deep neural networks proved that ANNs could now be trained at web-scale. These two paradigms view stories, and literature more broadly, through very different lenses. For instance, LLMs seemingly make no distinction between a text's surface and deep structures, and view literary generation as an end-to-end mapping of text inputs (a *prompt*) to text outputs (a *continuation*). Symbolic AI models, in contrast, typically identify different stages in story production, so as to explicitly model concerns such as plotting, characterization and surprise (Pérez y Pérez and Sharples, 2023).

To understand the future relationship of AI to literature, we must first learn from their past entanglements, from the purely symbolic to the statistical. A brief history of AI story generation is therefore presented in the next section. To understand how stories cultivate – and often dash – reader expectations, we next consider the problem of joke generation. Jokes, after all, are very short stories whose brevity encourages a great many short-lived expectations, and while LLMs currently exhibit a facile wit and a capacity for wordplay, it is instructive to explore why they still lack the ability to invent a good joke. Does this failing reflect the inherent difficulty of joke creation, or does it signify an intrinsic rigidity in how AIs model the expectations we bring to a text? A case study in the use of LLMs to generate comics is then presented, which allows us to offer some tentative predictions for the future of LLM-driven literary creativity.

2. What to expect when you're expecting … a story

Generation is the flipside of analysis. What we learn from analysis can be used to inform our generators, while a capable generator can demonstrate the validity of our analytical insights. Indeed, generation often yields more insights than pure analysis, insofar as simple mechanisms working in concert can give rise to complex effects that might otherwise suggest a theory with too many moving parts (Braitenberg, 1984). In the case of story generation, this branch of AI owes a significant debt to the early structural analyses of Propp (1928) and Campbell (1949). The former's inventory of morpheme-like story functions in a corpus of Russian folktales set the stage for combinatorial AI approaches to come (e.g., Gervás, 2016) while the latter's positing of a "monomyth" structure to ancient tales of heroic endeavour laid the ground for generation via story "grammars" (Rumelhart, 1980; Veale, 2017). Those scholars were not interested in generation *per se* – certainly not automated generation – but even in 1928 we see efforts to systematize the creation of commercial stories that owe much to the structuralists' insights, and which today look remarkably prescient in light of later developments in symbolic AI (Cook, 1928).

As such, story generation is a branch of AI – for the most part, *symbolic* AI – with a long provenance. Within this tradition, there are two high-level approaches to automating the story-writing process: one can model the goals and desires of a real author, or one can model the goals and desires of a fictional character. In contrast, LLMs tend to directly model the text itself as a polished artifact. This is not to say that LLMs cannot generate stories with compelling plots and believable characters, rather that these aspects are not explicitly treated in a dedicated, modular fashion that one can easily inspect.

Klein et al.'s (1973) *Novel Writer*, which is perhaps the first symbolic story generator of note, produced murder-mystery tales by simulating the "potential behavior of a set of characters." Meehan's (1977) TALE-SPIN would later generate tales of woodland creatures by first creating a world of inter-related characters for a story to selectively describe and follow. Dehn's (1981) AUTHOR system represents the earliest attempt to seriously model the mind of a writer, by focusing on the authorial goals that constrain the development of a story. Lebowitz's (1985) UNIVERSE system was designed to generate scripts for soap-operas, while Turner's (1993) MINSTREL system focused on a very different genre, tales of Arthurian knights. Turner was chiefly motivated by what had worked for human authors in the past, and so relied upon a set of *Transformer-Recall-Adapt* methods to guide his generator's choices. The MEXICA system of Pérez y Pérez and Sharples (2001) also generates tales of knights and princesses, albeit in an Aztec setting, but much like Dehn's

AUTHOR, it also aims to model the mental processes of a human writer. Its E-R model of Engagement and Reflection engages with a developing story by first adding new actions, perhaps by dipping into its memory of past stories, before the reflection phase then considers the ramifications of those choices.

Other symbolic approaches to story generation engage in simulations of characters and their emotions (e.g., the STellA system of León and Gervás (2014), or make use of structuralist principles to generate new stories from an established ontology of character functions. The *PropperWryter* system of Gervás *et al*. (2019) uses insights from Propp's analysis of Russian folk tales to generate its plots, and has contributed to the plot of a West-End musical (titled *Beyond The Fence*). Although partially machine-generated, the musical was performed by human actors, rather than robots. In contrast, the stories produced by the *Scéalability* system of Veale (2017) are given a physical embodiment through the gestures and movements of two performing NAO robots (Wicke and Veale, 2020). *Scéalability* follows what is, essentially, a story-grammar approach to narrative generation (Rumelhart, 1980); an extensive grammar of action triples is first used to generate the fabula or plot, before this skeletal structure is then rendered into idiomatic English. A dash of whimsy is added by way of a large knowledge-base of familiar characters, real and fictional, whose pairings in apt but unexpected ways also foster humorous incongruity (Veale, 2021).

Most symbolic story generators rely on schematic generalizations in the mould of Propp, Cook and Campbell, and model storytelling as a process of instantiating a set of generic, reusable forms in diverse ways. Alternately, a system may employ an even higher-level goal-driven process that yields these structures on the fly. Conventional wisdom holds that a story must have “ups” *and* “downs”, so that the protagonist of a story faces obstacles with non-trivial solutions on the path to achieving their aims. Since the plans constructed to overcome these obstacles form a key part of the plot, and reveal to us a great deal about the characters themselves, Riedl and Young (2010) exploit this symmetry to view storytelling as a planning process. In AI, a plan is a temporally ordered sequence of actions to be executed in the furtherance of a goal; by transplanting this sequence onto a tale’s characters, a symbolic planner can give purpose to their actions and substance to their beliefs.

3. What to expect when you're expecting … a joke

When our schematic generalizations about stories mesh together seamlessly, they usually work unnoticed in the backstage of cognition. It is only when they work against each other, as they so often do in jokes, that these predictive patterns and their limitations are thrown into the spotlight. Take this classic joke, which activates one time-worn pattern before ditching it for another:

> My grandpa died peacefully in his sleep.
>
> Unlike his passengers, who died screaming.

According to Raskin (1984), a text must be compatible, in part or in whole, with two conflicting scripts for it to exhibit a humorous potential. A "script" is Raskin's term – which he borrows from Schank and Abelson (1977) – for the causal generalizations that guide our understanding of a narrative. He speaks merely of *potential* because humour can be subjective, and different audiences may react unevenly to the same stimulus. In the joke above, a key script is evoked by the setup, leading us to imagine much more than we are told. We infer that grandpa passed away quietly in his own bed, until the punchline forces us to invalidate this inference and instead place grandpa behind the wheel of a passenger vehicle. Perhaps you imagine a bus full of screaming school kids, or a coach filled with shrieking holidaymakers? The joke does not tell us what to think, but it does engage our individual imaginations to vividly plug the gaps. This gap-filling also spurs us to morally re-profile the events of the joke, so that grandpa shifts from victim to villain. No longer the blameless sufferer of a heart attack, grandpa is now recast as a careless driver who falls asleep at the wheel. Our emotions shift too, from sympathy to blame, as we point the finger at him.

There is no logical force to this interpretation, which in the end is just one of many possible readings of the joke. Perhaps grandpa *did* die at home in his bed, and it was the substitute driver who led grandpa's regular passengers – "his" passengers as the joke puts it – to their doom? As noted in Veale (2004), jokes are not the logical mousetraps they are often portrayed to be. We are not trapped into thinking one way and then another, but merely invited to think in ways that maximize the humorous impact of the text. We actively collude in the telling of the joke by meeting the teller halfway. By placing a sleepy grandpa behind the wheel of a crowded passenger vehicle, we get to blame the sweet old man for the awful tragedy that ensues. Without this blame, the joke loses its sweet-and-sour tang and fails to elicit a chuckle.

The conflicting script is introduced with the contrast marker “Unlike.” This defies our expectations on one level, but respects them on another. After all, we expect jokes to toy with our expectations and to leverage them against us. Raskin’s original proposal from 1984, his Semantic Script Theory of Humour (or SSTH), is elegantly minimal in its formulation: it simply posits the existence of a large, open-ended set of conceptual scripts, which a given text can activate to differing degrees, and a much smaller set of semantic oppositions along which those scripts can generate friction. As in grandpa’s case, the oppositions that are most conducive to humour tend to elicit moral or social judgments from an audience, and include *good/bad*, *high/low stature, and obscene/non-obscene*. As the joke unfolds, it becomes clear that the initially activated script is no longer fit for purpose, and a second must be engaged to explain the latest twist. The opposition between the two then spurs the audience to search for a resolution, to explain why these incompatible scripts can work together in this special case. This inter-script friction – or *incongruity*, as scholars dub it – is not funny in itself (Veale, 2004), rather it is the process of sense-making that follows that we find so satisfying. The skill of the comic lies not in evoking an opposition, but in imagining the unusual circumstances that make it surprisingly meaningful.

The comedian’s skillset draws on other factors that fall outside the remit of the SSTH, from timing – which is crucial – to word choice and phrasing, narrative strategy (e.g., quip, aphorism, shaggy dog tale, narrative joke) to knowledge of the audience and shared knowledge of the joke’s target or *butt*. For these, Attardo and Raskin’s (1991) GTVH, or *General Theory of Verbal Humour*, erects a more modular theory on the foundations of the SSTH. The GTVH aims to explain how different jokes can emerge from the same modular mechanisms. The theory’s most profound addition is also its most Proppian: the idea that every joke instantiates a specific logical mechanism that provides a high-level rationale for how and why the two scripts are to be set in conflict. It is the chosen logical mechanism, such as *false analogy* or *faulty reasoning*, that allows a joke to appear nonsensical at one level and quite sensible at another.

While scripts, oppositions and logical mechanisms are semantic constructs that operate at different levels of abstraction, they must all work together in the end, just as one cog must mesh with others if a clock is to keep ticking. Since a common representation helps to ensure that this is the case, Raskin's conception of scripts has evolved from the SSTH's rigid, set-like structures into richer, more malleable graphs (Attardo, Hempelmann and Di Miao, 2002). With the GTVH's evolution into the *Ontological Semantics Theory of Humour,* or OSTH (Raskin, Hempelmann and Rayz, 2009), scripts, oppositions and logical mechanisms now reside in the same densely cross-indexed ontology. By extending the GTVH's Proppian turn, the OSTH has become closer in spirit to Cook's *Plotto*, with its practical focus on the large-scale connectivity of ideas. It is this connectivity that allows us to nimbly hop between viewpoints, so as to approach and reframe even the most familiar ideas in unexpected ways.

4. How To Expect The Unexpected

The expectations we bring to a story will range from the generic and abstract to the concrete and domain-specific. Although AI researchers and humanities scholars approach stories from very different angles, each has attempted to model these expectations in surprisingly similar ways. And while the diversity of our expectations has produced a corresponding diversity in the representations and processes that we build into our symbolic models, recent advances in the scaling and training of large language models offer a more unified view of the expectations we bring to the appreciation and generation of all kinds of stories.

A language model is, in essence, a probability distribution over the possible strings of a language. This allows an LLM to support a uniform probabilistic approach to the representation of every kind of linguistic expectation, from the paradigmatic to the syntagmatic and the semantic to the pragmatic. High-level questions of genre and script, and low-level questions of grammar and word choice, can all be uniformly answered by asking: what is the probability of any given word being the next word in an unfolding text? By attuning to context, and learning context-specific representations of the words in the training data, LLMs learn to master the diverse language games that we humans play with each other. But are LLMs more than the "stochastic parrots" that some (e.g., Bender et al., 2021) have labeled them? An analysis by Jentzsch and Kersting (2023) argues that ChatGPT is *fun* but not *funny*, in part because its LLM (GPT3.5T) has been over-exposed during training to a small set of 25 jokes. ChatGPT does well when asked to explain humorous metaphors and similes, and it can grasp the intent of witty insults. But when asked to generate its own jokes, the results are painfully meretricious in their reliance on puns, scare-quotes and clichés.

Nonetheless, the system improvises well in a comedic fashion. A foundational principle of *improv comedy* is "*yes, and* …": each performer tacitly says "*yes*" to the premises introduced by others "*and*" proceeds to build upon them. In this way, a silly premise becomes increasingly strained as it undergoes continued elaboration. Language models work in much the same way: given the initial premise of a user's prompt, an LLM auto-regressively unfurls a continuation that takes this premise at face value ("yes") and builds upon it ("and"). If the prompt plants the seed of a comic situation – for example, "Suppose they built a rotating restaurant on top of the leaning tower of Pisa" – the continuation will draw out its comic potential – "and the waiters serve hot soup while wearing roller skates." Unsurprisingly, tech-savvy improv performers (e.g., Mirowski et al., 2020) have jumped all over the hallucinatory tendencies of LLMs to do just that. This use of AI may be interesting if you like improv, but perhaps less so if you like comedy.

Before training, the only *a priori* elements of an LLM are its hyper-parameters. These fix the number, width and type of its stacked layers, the dimensionality of the vector embeddings that it learns for different words, the size of its context window (how many words it must attend to at once), and the number of attention heads it should concurrently apply to this context (Radford et al., 2019). As a bottom-up learner, an LLM evolves its own contextual representations for each word in each context it is used, and simultaneously learns which aspects of these representations to attend to in different settings. This gives LLMs a mastery of style and genre, but it also allows them to generalize over content. An LLM naturally interpolates between the observations in its training data to imagine – or *hallucinate* – coherent arrangements of words it has never seen, while its learnt representations allow it to generalize from observed patterns to analogous arrangements of similar words that are, in a real sense, truly novel. This makes LLMs capable of what Giora *et al*. (2004) dub *optimal innovation*, or the ability of creative writers to sensibly balance the novel with the familiar.

From a given prompt, which may well be absurd to begin with, an LLM will unfurl a text that obeys the tacit rules of whatever language game the user is playing. The coherence of the unfurled text is determined by the context window over which the LLM's attention heads work, which can range from hundreds to thousands of words. As the LLM auto-regressively adds each new word, this window slides one step forward, so that earlier elements gradually lose their influence over later additions. This loss of coherence is subtle at first, but the cracks widen as the window shifts further away from the text's initial premises. Consider a short story generated by the GPT-2 LLM as reported in Radford *et al*. (2019). The story unfurls in response to a prompt written in the breathless style of a tabloid newspaper, and continues in that genre in what seems like an homage to Raymond Queneau's *Exercises In Style* (1949). The LLM's tale recounts the discovery of English-speaking unicorns in the Andes, and goes on to speculate on the origins of this odd species, which it names "Ovid's unicorn" for its "distinctive horn." However, the species is said to be "four-horned" in the next sentence, but no further mention is made of this surfeit of horns as the text unfurls. Rather, true to its name, the unicorn later returns to being uni-horned. As befits an optimal innovation, the LLM strives to balance the novel with the familiar, but the familiar can sometimes stifle the most peculiar innovations.

The loss of coherence over long texts goes hand in hand with the user's own loss of control. To regain control and minimize contextual drift, we can divide an ambitious goal into smaller sub-tasks. If these are fed sequentially to an LLM, its piecewise outputs can then be re-assembled to form a single *Franken*-text, provided that each sub-task works within a unifying context that is passed from one task to the next. To test this approach, Ventura (2023) explored the capacity of ChatGPT (see Leiter *et al*. 2023), a version of GPT-3.5 fine-tuned for dialogue, to serve as a co-author in the writing of an academic paper. Ventura's end result is a paper within a paper: the inner paper was written with ChatGPT as a co-author, and explores the role of LLMs as co-creative writing partners, while the outer paper is a wholly human critique of this co-creative enterprise.

Starting from his own initial premise, Ventura generated a skeletal outline of the desired paper, and repeatedly prompted ChatGPT to put flesh on these bones. While the LLM did a good job of producing the kind of content that e.g., one expects of a literature review, it was robbed of any narrative momentum by the fractured nature of its interactions. The generated text fragments exhibited a great deal of overlap and repetition, and required considerable post-editing to work into a coherent whole. More seriously, ChatGPT "hallucinated" most of its bibliographic references, despite academic citation being one of the few literary practices that have no need of optimal innovation. In the end, ChatGPT *did* write a significant amount of the finished paper, but demanded so much effort – too much, in Ventura's view – to make such a collaboration worthwhile for now.

This is what we can expect whenever we impose, from the outside, a top-down organization onto an essentially bottom-up process. We might expect better results if we ask LLMs to bring their own internal top-down intuitions to bear, since LLMs are trained with very many examples of humans doing just that. Consider how an LLM might respond when prompted with a brainteaser that requires algebraic or logical reasoning. If we keep the numbers human-scale, LLMs will often give the right answer, but they are still prone to silly errors. A little prompt engineering, however, can improve the LLM's success rate and provide more edifying solutions. Just by asking the LLM to "work the problem logically" or "to show its reasoning," we can elicit outputs that exhibit top-down organization and a clear chain of thought. We are, in effect, prompting an LLM to play a different language game, one often played in textbooks, in which intermediate steps provide a clear and informative context for later ones.

In the same vein, the writer Joe Toplyn has turned his professional insights about jokes (Toplyn, 2014) into an AI system named *Witscript* (Toplyn, 2022) that guides an LLM along a chain of thought that is conducive to comedy. The system first prompts the LLM to identify the two main ideas, or "handles," in a topical headline, before prompting it to suggest angles on each that will show them to be related. The two handles are textual, but they implicitly designate scripts in the sense of Raskin's SSTH, while the angles designate their points of overlap. If the two scripts are sufficiently different, a natural opposition will suggest itself (Veale, 2004), providing the incongruity that so often fuels wit. Witscript's LLM of choice, the latest GPT model, already possesses a facile wit of its own, one that captures the tones and rhythms of office banter rather well even if it still lacks the ability to write new jokes to a professional standard (Jentzsch and Kersting, 2023). However, by imposing an explicit control sequence that exploits aspects of humour theory, Toplyn's system is able to elicit greater topicality and incongruity from its LLM. More generally, Witscript itself hints at the shape of things to come for LLMs in creative writing. If an LLM is to be our sorcerer's apprentice, we should not expect it to make leaps of the imagination, or jump to polished solutions in a single bound, but to serve as a cooperative partner in the step-by-step quest for better literary outputs.

5. Case Study: Generating Content for Topic Comic Strips

Like many paradigm shifts in science, the transition from symbolic to neuro-statistical AI has been neither sudden nor total. The two paradigms evolved side-by-side for decades, through the ups and downs of various AI winters, before advances in hardware and algorithms granted supremacy to the latter. But there is still much that a symbolic viewpoint can lend to the robustness and scalability of an LLM-based approach to literary creativity, not least the ability to guide the creation process along a top-down plan of action. When a task calls for symbolic knowledge to drive this process, we might even ask the LLM to produce this knowledge on demand, and in the desired symbolic format, since so much of what an LLM learns from web data is technical in nature.

As a case in point, consider the *Excelsior* system described in Veale (2022, 2023). Excelsior is a symbolic generator of topical comic strips that strives to present a balanced view of the chosen topic. An Excelsior comic is specified as a sequence of panels using a bespoke XML schema, which specifies each panel as a combination of character poses, background images, captions and speech balloons. The pipeline from topic to comic is wholly symbolic, and first utilizes a number of distinct knowledge resources to flesh out its topic as a set of propositions. Each of these is given its own panel in the comic, with two recurring characters who argue for and against the proposition. These opposing characters are posed to suit their dialogue in each panel, and are placed against an equally apt choice of background. To augment the system's own knowledge with dynamic content, a trawl of Twitter is also performed to identify trending hashtags on the given topic. A set of hashtag templates, such as *#{X}Sucks* and *#Saint{X}*, then maps any matching tags to an OSTH-like ontology to extract their propositional content and to give each an emotional framing. The ontology's topology allows these framings, and their panels, to be ordered by emotional intensity to form a simple plot. For instance, a debate may move from *curiosity* to *suspicion* to *disgust* and *hatred* on one side of its treatment of X.

This pipeline is a proficient piece of symbolic clockwork, with a great many decision points along the way that shape the substance of the final comic. Yet, despite its proficiency, it rarely delivers results that surprise and delight. Rather, it does what it is expected to do, with formulations that soon become stilted and stale with overuse. The system is the sum of its parts, and no more. This is true of an LLM also, but the breadth of its training data is so vast that no one person could ever internalize it all, or see the outputs of the LLM as an obvious function of a certain rule or a particular piece of prior knowledge. By replacing Excelsior's symbolic pipeline with an LLM, we may greatly simplify its processes while also enhancing its capacity for surprise and occasional delight.

To start, we obtain a propositional model of the topic by asking the LLM for a set of semantic triples, of the kind used to build a symbolic knowledge graph. So that the LLM fully understands our request, we provide it with an example:

User: A semantic triple of the form (Subject, predicate, object) that describes the concept Donut is as follows:

(Donut, has part, hole in the middle)

Now suggest 10 semantic triples to represent a neutral view of the concept Elon Musk.

ChatGPT: Here are 10 semantic triples about Elon Musk:

1. (Elon Musk, founder of, SpaceX)
2. (Elon Musk, CEO of, Tesla)
3. (Elon Musk, innovator in, transportation technology)
4. (Elon Musk, investor in, artificial intelligence)

: : :

10. (Elon Musk, believer in, colonizing Mars)

From these propositions we now generate the dialogue for a comic which, in this case, takes a balanced view of the topic *Elon Musk*. To achieve balance, we ask the LLM to match a benign view to a cynical view of each proposition in turn:

User: Pithily express these triples as rhyming couplets, pairing a benign view with a cynical view in each. Prefix each with benign: or cynical: as appropriate.

ChatGPT: Certainly.

1. Benign: Elon Musk, SpaceX founder, reaching for the stars.
   Cynical: Elon Musk, SpaceX founder, just wants to sell cars.
2. Benign: Elon Musk, Tesla's CEO, making green cars cool.
   Cynical: Elon Musk, Tesla's CEO, just wants to play the fool.
3. Benign: Elon Musk, transportation tech innovator, changing the game.
   Cynical: Elon Musk, transportation tech innovator, just wants the fame.
4. Benign: Elon Musk, AI investor, pushing the boundaries.
   Cynical: Elon Musk, AI investor, just wants more money.

: : :

10. Benign: Elon Musk, Mars colonization believer, exploring the unknown.
    Cynical: Elon Musk, Mars colonization believer, just wants to be shown.

We ask the LLM for interlocking couplets so to reinforce the sense of balance in each panel. By batching our request, so that It operates on all ten propositions at once, we also ensure stylistic coherence across the ten couplets that are returned. Notice how each has the same call-and-response format, which lends the resulting comic a consistent feel overall, somewhat in the vein of the *MAD Magazine* parodies of old. We now have pithy dialogue to fill the speech balloons of our opposing characters; we next need an independent observer to voice the panel captions. Each caption must speak to the proposition debated in a panel, and do so in a distinctive voice of an recognizable narrator. We can ask an LLM to mimic any individual whose speech patterns are widely-parodied on the web, such as Donald Trump, to heighten the comedic sensibility:

User: Pithily express these triples about Elon Musk as utterances in the style of Donald Trump. Use the same numbering as the triples.

ChatGPT:

1. Elon Musk, tremendous founder of SpaceX, folks.
   Believe me, nobody does space like him.
2. Elon Musk, fantastic CEO of Tesla.
   He's making America great again with electric cars.
3. Elon Musk, incredible innovator in transportation technology.
   He's doing a tremendous job, believe me.

4. Elon Musk, tremendous investor in artificial intelligence.
   Nobody does AI like him, believe me!

   : : :

10. Elon Musk, big believer in colonizing Mars.
    He's making America great again in space, folks.

With chaining and batching our requests to the LLM in this way, it is possible to replace the bulk of Excelsior's symbolic pipeline with a simple prompting model. Context is vital to each of the prompts above, and must be carefully maintained. We can, for instance, prime our initial prompt for semantic triples by establishing a prior context in which the LLM is asked to summarize a set of recent hashtags about the topic, as dynamically retrieved from Twitter. This technique, known as *Retrieval-Augmented Generation* (or RAG), allows the LLM to use knowledge or database content that falls outside its original training data (Lewis et al., 2020).

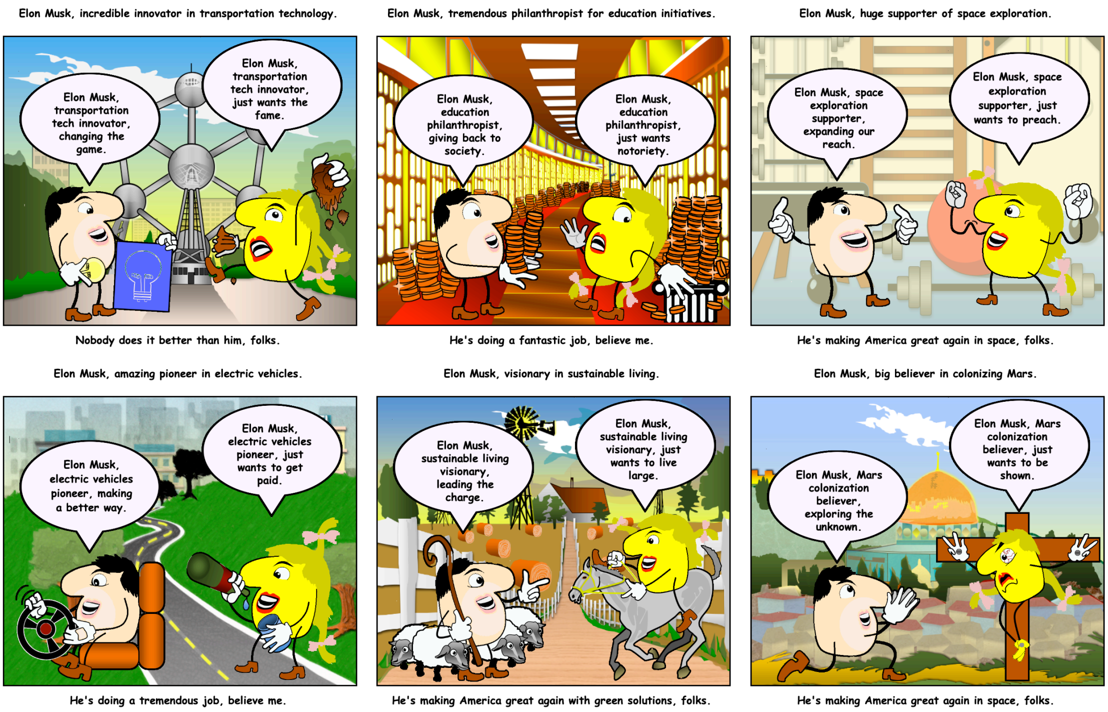

Figure 1. One page of an Excelsior/LLM comic on the topic of Elon Musk.

It now remains for us to visually render the substance of the comic, as produced by the LLM, as a comic strip. To choose an apt pose for each piece of dialogue, we match a vector-space embedding of the dialogue text to the embeddings of each of Excelsior's allowable poses. These vector embeddings, as provided by the makers of ChatGPT, allow nuanced similarity judgments to be made in the latent geometric space that lies beneath the brittle realm of words and symbols. A comparable mapping is made from each semantic triple to Excelsior's library of predefined backgrounds, to choose an apt background setting for each panel. A page of the resulting comic on the topic of Elon Musk is presented in Figure 1.

6. Conclusions: Symbolic Caterpillars and Statistical Butterflies

It is an oversimplification to state that LLMs reduce everything to probabilities. The probabilistic view is nonetheless useful when talking about expectations, and especially those of creative authors and their audiences. By assigning probabilities to different continuations of a prompt, an LLM allows us to quantify the extent to which each is expected, and the surprise it might evoke. The more expected the continuation, the higher the probability assigned to it. Humour, or optimal innovation more broadly, complicates this simple picture. An unexpected continuation must still make sense, and must still cohere with its initial prompt. A humorous continuation must have a lower probability, but not *too* low. Whenever we prompt an LLM to deliver creative – perhaps humorous – results, we are, in effect, urging it to generate *familiar surprises*: outputs that surprise us in context while still clearly echoing that context and the LLM's own training data.

Mimicry is a source of many familiar surprises. Human mimics do more than simply impersonate a famous target; they create as they mimic, to parody their target by inventing content that is both new and oddly familiar at the same time. This is why it is a mistake to dismiss LLMs as mere "stochastic parrots." As has been argued here, the better animal metaphor is perhaps "statistical butterflies." A single LLM is versatile enough to replace most of the components of a typical symbolic pipeline for literary generation, and proves a more nimble substitute in each case than the original bespoke component. LLMs bring context-sensitivity and nuance to each task they replace, allowing us to lend a distinctive voice to even the most mundane aspects of the generative process. So, in striving to exercise more control over this process in our automated literary endeavours, there is still value in using the top-down structures of existing symbolic systems as a means of reining in the wild generativity of an LLM, to deliver surprises that delight while being reassuringly familiar in all the ways that we want them to be.

References

Attardo, Salvatore, and Victor Raskin. (1991). Script theory revis(it)ed: joke similarity and joke representational model. *Humor: International Journal of Humor Research* 4, no. 3: 293-347.

Attardo, Salvatore, Christian F. Hempelmann, and Sara Di Maio. (2002). Script oppositions and logical mechanisms: Modeling incongruities and their resolutions. *Humor: International Journal of Humor Research* 15, no. 1: 3-46.

Bender, Emily, Timnit Gebru, Angelina McMillan-Major and Margaret Mitchell (2021). On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? *Proceedings of FAccT '21, the ACM Conference on Fairness, Accountability, and Transparency*, 610-621.

Braitenberg, Valentino. (1984). *Vehicles: Experiments in synthetic psychology*. Cambridge, MA: MIT Press.

Campbell, Joseph. (1949). *The Hero With a Thousand Faces*. New York: Pantheon Books.

Cook, William Wallace. (1928/2011). *PLOTTO: The Master Book of All Plots*. Portland, Oregon: Tin House Books.

Dehn, Natalie. (1981). Story generation after tale-spin. *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vancouver, Canada, vol. 1: 16–8.

Gervás, Pablo. (2016). Computational Drafting of Plot Structures for Russian Folk Tales. *Cognitive Computation* 8, no. 2:187–203.

Gervás, Pablo, Eugenio Concepción, Carlos León, Gonzalo Méndez and Pablo Delatorre. (2019). The long path to narrative generation. *IBM Journal of Research and Development* 63, no. 1: 1–9.

Giora, Rachel, Fein, Ofer, Kronrod, Ann, Elnatan, Idit, Shuval, Noah and Zur, Adi. (2004). Weapons of Mass Distraction: Optimal Innovation and Pleasure Ratings. *Metaphor and Symbol* 19, no. 2: 115–141.

Jentzsch, Sophie and Kersting, Kristian. (2023). ChatGPT is fun, but it is not funny! Humor is still challenging Large Language Models. *ArXiv* 2306.04563.

Lebowitz, Michael. (1985). Story-telling as planning and learning. *Poetics* 14, no. 6: 483–502.

Leiter, Christoph, Zhang, Ran, Chen, Yanran, Belouadi, Jonas, Larionov, Daniil, Fresen, Vivian and Eger, Steffen. (2023). ChatGPT: A Meta-Analysis after 2.5 Months. *ArXiv* 2302.13795.

León, Carlos and Pablo Gervas. (2014). Creativity in story generation from the ground up: Nondeterministic simulation driven by narrative. *Proceedings of the 5th International Conference on Computational Creativity*, Ljubljana, Slovenia, 201–210.

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel and Douwe Kiela. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Meehan James R. (1977). Tale-spin, an interactive program that writes stories. *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, Cambridge, MA, vol. 1: 91–98.

Mirowski, Piotr, Kory Mathewson, Boyd Branch, Thomas Winters, Ben Verhoeven and Jenny Elfving. (2020). Rosetta Code: Improv in Any Language. *Proceedings of the 11th International Conference on Computational Creativity*, 115–122.

Pérez y Pérez, Rafael and Sharples, Mike. (2001). MEXICA: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence* 13, no. 2: 119–139.

Pérez y Pérez, Rafael and Sharples, Mike. (2023). *An Introduction to Narrative Generators: How Computers Create Works of Fiction*. London, UK: Oxford University Press.

Propp, Vladimir. (1928/1957). *Morphology of the Folktale*. Bloomington, Indiana: American Folklore Society.

Queneau, Raymond (trans: Barbara Wright). (1947/2012). *Exercises in Style*. New York: New Directions Publishing.

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei and Ilya Sutskever. (2019). Language models are unsupervised multitask learners.

OpenAI blog 1, no. 8: 9.

Raskin, Victor. (1984). *Semantic Mechanisms of Humor*. Dordrecht: D. Reidel.

Raskin, Victor, Christian F. Hempelmann, and Julia M. Rayz. (2009) How to understand and assess a theory: The evolution of the SSTH into the GTVH and now into the OSTH*. Journal of Literary Theory* 3, no. 2: 285-311.

Riedl, Mark and Michael Young. (2010). Narrative planning: balancing plot and character. *Journal Artificial Intelligence Research (JAIR),* 39:217–268.

Rumelhart, David E. (1980). On Evaluating Story Grammars. *Cognitive Science* 4, no. 3: 313–316.

Schank, Roger, and Robert P. Abelson. (1977). *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures*. New Jersey: Lawrence Erlbaum Associates.

Toplyn, Joe. (2014). *Comedy Writing for Late-Night TV*. New York: Twenty Lane Media.

Toplyn, Joe. (2022). Witscript 2: A System for Generating Improvised Jokes Without Wordplay. *Proceedings of ICCC 2022, the 13th International Conference on Computational Creativity*. Bolzano, Italy, June.

Turner Scott. R. (1993). Minstrel: a computer model of creativity and storytelling. PhD thesis, University of California at Los Angeles, Los Angeles, CA, USA.

Veale, Tony. (2004). Incongruity in Humor: Root-Cause or Epiphenomenon? *Humor*: *The International Journal of Humor Research* 17, no. 4: 419–428.

Veale, Tony. (2017). Déjà Vu All Over Again: On the Creative Value of Familiar Elements in the Telling of Original Tales. *Proceedings of the 8th International Conference on Computational Creativity*, Atlanta, Georgia.

Veale, Tony. (2021). *Your Wit Is My Command: Building AIs With a Sense of Humor*. Cambridge, MA: MIT Press.

Veale, Tony. (2022). Two-Fisted Comics Generation: Comics as a Medium and as a Representation for Creative Meanings. *Proceedings of the 13th international conference on Computational Creativity. Bolzano, Italy*, 59–66.

Veale, Tony. (2023). Have I Got Views For You! Generating “Fair and Balanced”

Interventions into Online Debates. *Proceedings of the 14th international conference on Computational Creativity*, Waterloo, Canada, 2–9.

Ventura, Dan. (2023). The Emperor's New Co-Author. *Proceedings of ICCC 2023, the 14th International Conference on Computational Creativity*. Waterloo, Canada, 55–63.

Wicke, Philipp and Tony Veale. (2020). The show must go on: On the use of embodiment, space and gesture in computational storytelling. *New Generation Computing* 38, no. 4: 565–592.